# A Self-learning Template Approach for Recognizing Named Entities from Web Text

**Qian Liu[†‡], Bingyang Liu[†‡], Dayong Wu[‡], Yue Liu[‡], Xueqi Cheng[‡]**
[†]University of Chinese Academy of Sciences, Beijing, China
[‡]Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China
{liuqian, liubingyang}@software.ict.ac.cn
{wudayong, liuyue, cxq}@ict.ac.cn

## Abstract

Recognizing Chinese Named entities from the Web is challenging, due to the lack of labeled data and differences between Chinese and English. We propose a semi-supervised approach which leverages seed entities and the large unlabeled data to learn templates. Some high-quality templates are generated iteratively to extract new named entities based on the model of quality metrics. Experimental results show that our approach significantly outperforms the baseline method and it is robust against the changes of the Web.

## 1 Introduction

Compared to English Named Entity Recognition (NER), Chinese NER is more difficult. For example, although capitalization plays an very important role in English NER, this language-specific feature is useless in Chinese NER. Moreover, the lack of space between words in Chinese often makes the work based on word segmentation (Sun et al., 2002) failure. Especially, there are a lot of new words and ambiguous words in the Web text, which increase the error of word segmentation. The loss of precision propagates to NER.

Currently, NER is mainly based on supervised models. Such models perform well in single domain (Wang, 2009; Du et al., 2010). Unfortunately, the data of the Web is open-domain and always changing. The performance of them degrades badly, since the distribution of the Web data is different from that of the training data. For instance, the average F1 score of the Stanford NER, which is trained on the CoNLL03 shared task data set and achieves state-of-the-art performance on that task, drops from 90.8% to 45.8% on tweets (Liu et al., 2011). Despite a high-quality training data set, which has the same distribution as the Web data and covers all kinds of domains, can improve the performance of NER, as far as we know, there are no such labeled data. Furthermore, named entities change over the time, especially person names and company names. (Tsuchiya et al., 2009) shows that 20%∼30% of named entity types are replaced with new ones every year in Mainichi Newspaper articles. Such change is even more obvious in the Web data and leads to the unreliable labeled data. Constantly annotating new data is time-consuming and expensive.

In this paper, we propose a semi-supervised approach that uses self-learning templates to solve above problems. Instead of annotating a massive amount of data, we leverage a small number of named entities and the large unlabeled data to discover new named entities that can not be identified using the training data. Experiments show that our approach raises the F1 from 75.9% to 88.6% on the Chinese Web data without retraining the existing model, and it is robust against the changes of the data. Since no language-specific knowledge is used, our approach can easily be extended to other languages.

## 2 Related Work

There have been many approaches proposed to solve the problem of the lack of annotated data. (Wu et al., 2009; Chiticariu et al., 2010) focus on domain adaptation, which aims to reuse the knowledge among different domains. (Ling and Weld, 2012; Rüd et al., 2011; Han and Zhao, 2010) leverage information from external knowledge sources, such as Wikipedia, WordNet and search engine, to compensate for the insufficient training data. Some work builds crowdsourcing services to label data by human. For example, Amazon Mechanical Turk[1] provides a platform to obtain data in various domains such as email

[1] https://www.mturk.com/mturk/welcome



*International Joint Conference on Natural Language Processing*, pages 1139–1143,
Nagoya, Japan, 14-18 October 2013.

(Lawson et al., 2010), medicine (Yetisgen-Yildiz et al., 2010) and Twitter (Finin et al., 2010).

The work based on context templates is more closely related to our approach. (Etzioni et al., 2005) extracts named entities via domain-specific templates, which are learned from predefined templates. (Whitelaw et al., 2008) builds training data utilizing templates generated by millions of seeds. Although context templates are used to improve the perform of NER in our approach, they are learned automatically from the unlabeled data and we only use several seed entities.

## 3 Approach based on self-learning Templates

### 3.1 Overall Framework of Our Approach

The main idea of our approach is that learning the high-quality templates in the bootstrapping process.

The details are shown in Algorithm 1, where the pair $\langle name, type \rangle$ represents an entity, named $name$, in class $type$. # denotes a placeholder for entity. The substring, like $t_{s-2}t_{s-1}name_s^{(i)}t_{s+1}t_{s+2}$, denotes the $i^{th}$ NE in the set of seed entities and its context of four tokens long. First, for each entity $e$ in $E_{seed}$, we find sentences containing $e$ and create a temporary set of templates. Second, for each candidate template, we relocate all possible named entities $E_{temp}$ in $C_{corpus}$. Third, computing the quality of templates and adding the high-scoring ones into template set $TS_{template}$. Lastly, we compute the confidence of candidate entities that are generated in above process, and remove the entities whose confidence are below the threshold from the set of entities. The value of threshold will be discussed in Section 4.

### 3.2 Features of Template

Given a candidate template, we define three statistical features to measure the quality of it.

**effectiveness ($f_1$):** This feature reflects whether a candidate template is prone to mistakes. We assume that tokens outside of $E_{seed}$ are not named entities. It is a reasonable assumption in practice, because the loss caused by assumption will become lower and lower with the increase of $E_{seed}$. The effectiveness is calculated as

$$f_1(T, c) = p(e|T_c) \cdot p(T_c) = \frac{\#(correct\ e|T_c)}{\sum_i \#(e|T_c^{(i)})} \quad (1)$$

**Algorithm 1** Framework of Templates Learning

**Input:** a set of NEs: $E_{seed} = \{< name, type >\}$ ; unlabeled web pages: $Corpus$
**Output:** $E_{seed}; TS_{template}$
Initialize $C_{corpus}$: $\phi$
Initialize $TS_{candidate}$: $\phi$
Initialize $TS_{template}$: $\phi$
**while** $E_{seed}$ keep growing (or below the predefined number of loops) **do**
  **for** each $entity^{(i)} =< name^{(i)}, type^{(i)} >\in E_{seed}$ **do**
    Add all sentences containing $name^{(i)}$ to $C_{corpus}$
    Create templates $< t_{-2}t_{-1}\#t_{+1}t_{+2}, type >$ when the substring $t_{s-2}t_{s-1}name_s^{(i)}t_{s+1}t_{s+2}$ belongs to some sentence in $C_{corpus}$ and add them to $TS_{candidate}$
  **end for**
  **for** each candidate template $T^{(i)} =< t_{-2}^{(i)}t_{-1}^{(i)}\#t_{+1}^{(i)}t_{+2}^{(i)}, type >\in TS_{candidate}$ **do**
    Extract all matching tokens $< token, type >$ while the substring $t_{-2}^{(i)}t_{-1}^{(i)}token_{+1}^{(i)}t_{+2}^{(i)}$ belongs to some sentence in $C_{corpus}$ and add them to $E_{temp}$
  **end for**
  **for** each template $T^{(i)} \in TS_{candidate}$ **do**
    Calculate the score $(T^{(i)}, score)$ $= evaluation(TS_{candidate}|C_{corpus}, E_{seed}, E_{temp},)$
    **if** score$> \delta$ **then**
      Add $T^{(i)}$ to $TS_{template}$
    **end if**
  **end for**
  Find new candidate NEs $E_{candidate}$ using $TS_{template}$ from the remaining unlabeled data
  Select high-quality NEs: $E'_{seed} = filter(E_{candidate}|TS_{template})$
  Update: $E_{seed} = E_{seed} \cup E'_{seed}$
**end while**

where $\#(e|T_c^{(i)})$ denotes the number of correct entities extracted by the $i^{th}$ template in class $c$.

**discrimination ($f_2$):** This feature is used for measuring how close a candidate template is related to a class. The value is computed as

$$f_2(T, c) = tf(T, c) \cdot \left(1 + \log \frac{\#C}{1 + \#c_j}\right) \quad (2)$$

where $tf(T, c)$ denotes the normalized frequency of template, that is divided by the maximum frequency, $\#C$ is the number of classes in $E_{seed}$, and $\#c_j$ is the number of classes that template $T$ appears.

**diversity ($f_3$):** The more different and correct NEs in a class extracted by template $T$, the more likely other good NEs within the same class will be extracted by it. This feature is computed as

$$f_3(T, c) = \frac{\#\{(correct\ e|T_c) \wedge (different\ e|T_c)\}}{\#(correct\ e|T_c)} \quad (3)$$

### 3.3 Quality Metrics Model of Templates

Since the proposed features are not independent of each other, we propose an approach in which the value of a feature is adapted according to the values of other features. Although it is similar to (Wei et al., 2010), we improve it by only updating a part of templates to reduce the computational cost.

Formally, given a set of candidate templates $TS_{candidate} = \{T^{(1)}, T^{(2)}, ..., T^{(n)}\} \subset R^m$ , let $f_k : TS_{candidate} \rightarrow R$ denote the ranking function on the $k^{th}$ feature, where $f_k \in F = \{f_1, f_2, f_3\}$. Our goal is to combine all features to produce ranking list that are better than any individual feature and then return the templates with high ranking scores.

Following the traditional manifold ranking process (Zhou et al., 2004) with one ranking function. 1) Defining the similarity matrix $W$ on the template set $TS_{candidate}$: $W_{ij} = similarity(T^{(i)}, T^{(j)})$. 2) Symmetrically normalizing $W$ by $S = D^{-1/2}WD^{-1/2}$ in which $D$ is the diagonal matrix with (i, i)-element equal to the sum of the $i^{th}$ row of $W$. 3) Iterating $F(t+1) = \alpha SF(t) + (1-\alpha)F(0)$ until a global stable state, where $\alpha$ is trade-off parameter in $(0, 1)$, $F(0)$ denotes the initial ranking results and $F(t)$ denotes the ranking results of the $t^{th}$ round.

For two ranking functions $f_1$ and $f_2$, the ranking score of $f_1$ will be changed after combining the ranking score of $f_2$. Considering the cost of consistency both the ranking results in initial $f_1$ and the feedback from $f_2$, we define the cost function caused by refining $f_1$ with $f_2$ in the $(t+1)^{th}$ round iteration as

$$\varphi(f_1|f_2) = \frac{1}{2}\left(\sum_{i,j=1}^{n} W_{ij}\left\|\frac{1}{\sqrt{D_{ii}}}f_1^{(t+1)}(T^{(i)}) - \frac{1}{\sqrt{D_{jj}}}f_2^{(t)}(T^{(i)})\right\|^2 + \mu\sum_{i=1}^{n}\left\|f_1^{(t+1)}(T^{(i)}) - f_1^{(0)}(T^{(i)})\right\|^2\right) \quad (4)$$

where $f_1^{(0)}$ denotes the initial ranking scores of $f_1$. Let the best refined ranking score is $f^*$, we have

$$\frac{\partial}{\partial f_1}\varphi(f_1|f_2)\,|_{f_1=f^*} = f^* - S\cdot f_2^{(t)} + u(f_1^* - f_1^{(0)}) = 0 \quad (5)$$

$$f^* = \frac{1}{1+\mu}\cdot S\cdot f_2^{(t)} + \frac{\mu}{1+\mu}\cdot f_1^{(0)} \quad (6)$$

Let $\alpha = \frac{1}{1+\mu}$, then we have

$$f^* = \alpha\cdot S\cdot f_2^{(t)} + (1-\alpha)\cdot f_1^{(0)} \quad (7)$$

Therefore, we can iteratively compute the ranking scores in the $(t+1)^{th}$ round to find the best $f^*$ shown as follows, which is proven to be convergent (Wei et al., 2010).

$$f_1^{(t+1)} = \alpha\cdot S\cdot f_2^{(t)} + (1-\alpha)\cdot f_1^{(0)} \quad (8)$$

In our case, due to the templates with low ranking scores on $f_2$ are helpless to refine the results ranked on $f_1$. We omit the templates in $f_2$ that fall below the threshold and feedback the rest templates, signaled by $Top\_f_2$, to $f_1$. Then, the normalized matrix $S$ can be simplified as a block matrix $\begin{pmatrix} S_{Top\_f_2} & 0 \\ 0 & 0 \end{pmatrix}$. The equation above can be rewritten as

$$f_1^{(t+1)}(Top\_f_2) = \alpha\cdot S_{Top\_f_2}\cdot f_2^{(t)}(Top\_f_2) + (1-\alpha)\cdot f_1^{(0)}(Top\_f_2) \quad (9)$$

Moreover, if we only improve the identical templates, the similarity matrix $W_{Top\_f_2}$ and normalized matrix $S_{Top\_f_2}$ degrade to identity matrices, the final iteration equation is

$$f_1^{(t+1)}(Top\_f_2) = \alpha\cdot f_2^{(t)}(Top\_f_2) + (1-\alpha)\cdot f_1^{(0)}(Top\_f_2) \quad (10)$$

### 3.4 Confidence of Candidate NEs

There may be still noisy in the set of candidate NEs despite using the high-quality templates to find new entities. Therefore, we also need to filter out the non-NEs for the further processing to insure the high accuracy. This section corresponds to the line 19 of Algorithm 1. We utilize the pointwise mutual information (PMI) (Etzioni et al., 2005) to measure the closeness between extracted NEs and templates.

Given an extracted named entity $e$ and a template $T$, the PMI score is computed as

$$PMI(e, T) = \frac{Hits(e+T)}{Hits(e)} \quad (11)$$

where $Hits(\cdot)$ denotes the number of sentences searched in the whole unlabeled corpus.

The confidence of $e$ extracted by template $T_c$ belonging to class $c$ can be expressed as

$$confidence(entity, c) = \frac{1}{\#T_c}\sum_i PMI(entity, T_c^{(i)}) \quad (12)$$

where $\#T_c$ denotes the number of templates in class $c$.

## 4 Experiments

### 4.1 Data Set

We conducted experiments on a real data set collected from the Web, which is from August $1^{th}$ 2012 to August $31^{th}$ 2012[2]. The details of the data set are given in Table 1. Two fine-grained categories of *Person* are considered: *Singer* and *Athlete*. Note that the NER on fine-grained categories is more difficult than that on coarse-grained

[2]During the preprocessing step, HTML tags and ads are eliminated from the data.

categories such as *Person*, *Location* and *Organization*. We randomly selected 8,955 sentences and manually labeled them as test data, which contains 232 singers and 1,807 athletes. The labeled data is further split into two parts: one for training the baseline system and the other for testing. The test data set includes 65 singers and 406 athletes. We trained a linear CRF model (Lafferty et al., 2001) as the baseline using the BILOU scheme (Ratinov and Roth, 2009).

| | News | Forum | Microblog |
|---|---|---|---|
| Number | 1,087,926 | 420,278 | 49,037,301 |

Table 1: The composition of the data set.

### 4.2 Experimental Analysis

Table 2 gives some examples of learned templates. Using the learned templates and extracted named entities, an additional recognizer can be built very easily. It can discover some new named entities that are left out by models trained on labeled data. We performed experiments to make comparison between baseline system and AD_NER system, which integrates additional recognizer into the CRF model. The results are shown in Table 3∼5, where $p$ is the threshold of confidence score of candidate entity.

| Template | Q_score | Num. of NEs |
|---|---|---|
| *Singer* | | |
| 欢、#、庾(Huan, #, Geng) | 0.9000 | 60 |
| #演唱(sing) | 0.8109 | 279 |
| 天后#，(diva) | 0.6620 | 319 |
| 听着#的歌(listen to the song) | 0.6120 | 9 |
| 演唱#的歌(sing a song) | 0.5820 | 16 |
| *Athlete* | | |
| 冠军#，(champion) | 0.9059 | 2340 |
| 名将#，(famous athlete) | 0.8711 | 481 |
| 选手#，(player) | 0.8382 | 2440 |
| 战胜#夺冠(win) | 0.7724 | 123 |
| 选手#在比(in a competition) | 0.6423 | 274 |

Table 2: Examples of templates and their qualities (Q_score) and the number of extracted named entities.

| | Precision | Recall | F1 |
|---|---|---|---|
| Baseline | 93.1 | 41.5 | 57.4 |
| AD_NER($p = 0.001$) | 71.3 | 83.1 | 76.8 |
| AD_NER($p = 0.01$) | 75.4 | 88.4 | **81.4** |
| AD_NER($p = 0.1$) | 90.6 | 47.7 | 62.5 |

Table 3: Results on *Singer*.

| | Precision | Recall | F1 |
|---|---|---|---|
| Baseline | 97.8 | 65.9 | 78.8 |
| AD_NER($p = 0.001$) | 79.9 | 91.5 | 85.3 |
| AD_NER($p = 0.01$) | 92.8 | 88.3 | **90.5** |
| AD_NER($p = 0.1$) | 93.9 | 71.4 | 81.1 |

Table 4: Results on *Athlete*.

| | Precision | Recall | F1 |
|---|---|---|---|
| Baseline | 97.3 | 62.3 | 75.9 |
| AD_NER($p = 0.001$) | 76.2 | 90.0 | 82.5 |
| AD_NER($p = 0.01$) | 91.3 | 86.1 | **88.6** |
| AD_NER($p = 0.1$) | 93.2 | 67.8 | 78.5 |

Table 5: Overall experimental results.

From Table 3 and Table 4, we find the best performance of baseline on *Singer* is 57.4%, 21.4% lower than that on *Athlete*. This can be explained as the scale of labeled data impacts the performance of supervised method, because, in Table 1, the instances of *Singer* is not as sufficient as *Athlete*. However, the performance of our approach in the two categories remains stable. This illustrates that the large-scale unlabeled data is useful in the case of a lack of training data.

As shown in Table 5, the best F1 of AD_NER is 88.6%, which is 12.7% higher than F1 of baseline. Although there is a somewhat loss of precision, we obtain a large number of named entities. Moreover, the cost of our approach is lower than automatic annotation, since we do not need to retrain the supervised model. It is more effective when labeled data is complex and hard to construct while unlabeled data is abundant and easy to access. In the practice, our proposed approach can easily remain up-to-date and extend the well-trained supervised model without fine-tuning or any human intervention.

We further find that the overall precision of AD_NER with threshold at 0.1 is only 1.9% higher than that with threshold at 0.01, but the loss of recall is 18.3%. For this reason, the threshold can be set to 0.01.

## 5 Conclusion

In this paper, we propose an approach to build an additional named entity recognizer that can assist the existing supervised models. The experimental results on the real data set from the Web show that our method improves the F1 score from 75.9% to 88.6%.

## References


Laura Chiticariu, Rajasekar Krishnamurthy, Yunyao Li, Frederick Reiss, and Shivakumar Vaithyanathan. 2010. Domain adaptation of rule-based annotators for named-entity recognition tasks. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 1002–1012.

Junwu Du, Zhimin Zhang, Jun Yan, Yan Cui, and Zheng Chen. 2010. Using search session context for named entity recognition in query. In *Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval*, pages 765–766.

Oren Etzioni, Michael Cafarella, Doug Downey, Ana-Maria Popescu, Tal Shaked, Stephen Soderland, Daniel S. Weld, and Alexander Yates. 2005. Unsupervised named-entity extraction from the web: An experimental study. *Artificial Intelligence*, 165(1):91–134.

Tim Finin, Will Murnane, Anand Karandikar, Nicholas Keller, Justin Martineau, and Mark Dredze. 2010. Annotating named entities in twitter data with crowdsourcing. In *Proceedings of the NAACL HLT 2010 Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk*, pages 80–88.

Xianpei Han and Jun Zhao. 2010. Structural semantic relatedness: a knowledge-based method to named entity disambiguation. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 50–59.

John Lafferty, Andrew McCallum, and Fernando CN Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. *Proceedings of the Eighteenth International Conference on Machine Learning*, pages 282–289.

Nolan Lawson, Kevin Eustice, Mike Perkowitz, and Meliha Yetisgen-Yildiz. 2010. Annotating large email datasets for named entity recognition with mechanical turk. In *Proceedings of the NAACL HLT 2010 Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk*, pages 71–79.

Xiao Ling and Daniel S Weld. 2012. Fine-grained entity recognition. In *Proceedings of the 26th Conference on Artificial Intelligence*.

Xiaohua Liu, Shaodian Zhang, Furu Wei, and Ming Zhou. 2011. Recognizing named entities in tweets. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 359–367.

Lev Ratinov and Dan Roth. 2009. Design challenges and misconceptions in named entity recognition. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning*, pages 147–155.

Stefan Rüd, Massimiliano Ciaramita, Jens Müller, and Hinrich Schütze. 2011. Piggyback: Using search engines for robust cross-domain named entity recognition. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 965–975.

Jian Sun, Jianfeng Gao, Lei Zhang, Ming Zhou, and Changning Huang. 2002. Chinese named entity identification using class-based language model. In *Proceedings of the 19th international conference on Computational linguistics*, pages 1–7.

Masatoshi Tsuchiya, Shoko Endo, and Seiichi Nakagawa. 2009. Analysis and robust extraction of changing named entities. In *Proceedings of the 2009 Named Entities Workshop: Shared Task on Transliteration*, pages 161–167.

Yefeng Wang. 2009. Annotating and recognising named entities in clinical notes. In *Proceedings of the ACL-IJCNLP 2009 Student Research Workshop*, pages 18–26.

Furu Wei, Wenjie Li, and Shixia Liu. 2010. irank: A rank-learn-combine framework for unsupervised ensemble ranking. *Journal of the American Society for Information Science and Technology*, 61(6):1232–1243.

Casey Whitelaw, Alex Kehlenbeck, Nemanja Petrovic, and Lyle Ungar. 2008. Web-scale named entity recognition. In *Proceedings of the 17th ACM conference on Information and knowledge management*, pages 123–132.

Dan Wu, Wee Sun Lee, Nan Ye, and Hai Leong Chieu. 2009. Domain adaptive bootstrapping for named entity recognition. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 1523–1532.

Meliha Yetisgen-Yildiz, Imre Solti, Fei Xia, and Scott Russell Halgrim. 2010. Preliminary experience with amazon's mechanical turk for annotating medical named entities. In *Proceedings of the NAACL HLT 2010 Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk*, pages 180–183.

Dengyong Zhou, Olivier Bousquet, Thomas Navin Lal, Jason Weston, and Bernhard Schölkopf. 2004. Learning with local and global consistency. *Advances in neural information processing systems*, 16:321–328.